AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

1-9. (canceled)

10. (currently amended) A method of software loading and initialization in a distributed network of nodes, the method comprising:

persistently storing, in a first storage of a master node, a plurality of software packages and a plurality of boot images, wherein the plurality of software packages and the plurality of boot images will be used by the nodes in the distributed network;

persistently storing, in a second storage of the master node, software version information and node type information for each node in the distributed network;

receiving, at the master node, a request for a boot image and software packages from a node, in the distributed network, that is performing an initial boot;

based on the request, the master node determining software version information of the node to retrieve from the second storage;

the master node retrieving the software version information of the node from the second storage;

the master node determining, based on the software version information of the node, a boot image of the plurality of boot images and one or more software packages of the plurality of software packages to extract from the first storage;

the master node extracting the boot image and the one or more software packages from the first storage;

the master node delivering, to the node, the boot image and the one or more software packages;

wherein said node:

(a) stores the boot image and the one or more software packages in its local persistent storage,

(b) extracts software version information from the one or more software packages and stores the software version information in the local persistent storage,

(c) reboots and executes the boot image stored in the local persistent storage, and

(d) in response to executing the boot image, verifies the software version information with said master node by sending the software version information to the master node;

the master node receiving the software version information from the node;

in response to receiving the software version information, the master node determining whether the node has the correct software versions;

in response to the master node determining that [[if]] said node does not have the correct software versions, then the master node retrieving correct software packages from the first storage and sending the correct software packages to said node, wherein said node stores the correct software packages in the local persistent storage and completes booting by executing the correct software packages stored in the local persistent storage.

11. (canceled)

12. (previously presented) A method as recited in Claim 10, wherein said node, based on a command from said master node, does not store the one or more software packages in its local persistent storage device, allowing said master node to download test software

packages to said node and temporarily run said node using the test software packages, and wherein when said node reboots, the test software packages will no longer exist on said node.

13. (canceled)

14. (previously presented) A method as recited in Claim 10, wherein if said node has the correct software versions, then said node completes booting by executing the one or more software packages stored in the local persistent storage.

15. (canceled)

16. (original) A method as recited in Claim 10, wherein the master node has the ability to categorize nodes into classes where all of the nodes in a particular class of nodes have the same software configuration and may have differing processor types.

17. (previously presented) A method as recited in Claim 10, wherein each of the one or more software packages contains version information, dependency information, and other metadata information pertaining to software in the package.

18. (previously presented) A method as recited in Claim 10, wherein the boot image is customized for a particular type of node and provides basic low-level communications.

19. (previously presented) A method as recited in Claim 10, further comprising:

executing a composite image, that is installed by a user onto said master node, to create a subset of the plurality of boot images, a subset of the plurality of software packages, and node information; and

placing the subset of the plurality of boot images and the subset of the plurality of software packages in the first storage and the node information in the second storage.

20. (previously presented) A method as recited in Claim 10, wherein the master node retrieving the software version information creates the software version information from the second storage based on functional features included in the request.

21. (currently amended) A computer-readable storage medium carrying one or more sequences of instructions for software loading and initialization in a distributed network of nodes, which instructions, when executed by one or more processors, cause the one or more processors to perform:

persistently storing, in a first storage of a master node, a plurality of software packages and a plurality of boot images, wherein the plurality of software packages and the plurality of boot images will be used by the nodes in the distributed network;

persistently storing, in a second storage of the master node, software version information and node type information for each node in the distributed network;

receiving, at the master node, a request for a boot image and software packages from a node, in the distributed network, that is performing an initial boot;

based on the request, the master node determining software version information of the node to retrieve from the second storage;

the master node retrieving the software version information of the node from the second storage;

the master node determining, based on the software version information of the node, a boot image of the plurality of boot images and one or more software packages of the plurality of software packages to extract from the first storage;

the master node extracting the boot image and the one or more software packages from the first storage;

the master node delivering, to the node, the boot image and the one or more software packages,

wherein said node:

(a) stores the boot image and the one or more software packages in its local persistent storage,

(b) extracts software version information from the one or more software packages and stores the software version information in the local persistent storage,

(c) reboots and executes the boot image stored in the local persistent storage, and

(d) in response to executing the boot image, verifies the software version information with said master node by sending the software version information to the master node;

the master node receiving the software version information from the node;

in response to receiving the software version information, the master node determining whether the node has the correct software versions;

in response to the master node determining that [[if]] said node does not have the correct software versions, then the master node retrieving correct software packages from the first storage and sending the correct software packages to said node, wherein

said node stores the correct software packages in the local persistent storage and completes booting by executing the correct software packages stored in the local persistent storage.

22. (canceled)

23. (previously presented) A computer-readable storage medium as recited in Claim 21, wherein said node, based on a command from said master node, does not store the one or more software packages in its local persistent storage device, allowing said master node to download test software packages to said node and temporarily run said node using the test software packages, and wherein when said node reboots, the test software packages will no longer exist on said node.

24. (canceled)

25. (previously presented d) A computer-readable storage medium as recited in Claim 21, wherein if said node has the correct software versions, then said node completes booting by executing the one or more software packages stored in the local persistent storage.

26. (canceled)

27. (previously presented) A computer-readable storage medium as recited in Claim 21, wherein the master node has the ability to categorize nodes into classes where all of the

nodes in a particular class of nodes have the same software configuration and may have differing processor types.

28. (previously presented) A computer-readable storage medium as recited in Claim 21, wherein each of the one or more software packages contains version information, dependency information, and other metadata information pertaining to software in the package.

29. (previously presented) A computer-readable storage medium as recited in Claim 21, wherein the boot image is customized for a particular type of node and provides basic low-level communications.

30. (previously presented) A computer-readable storage medium as recited in Claim 21, further comprising:

executing a composite image, that was installed by a user, to create a subset of the plurality of boot images, a subset of the plurality of software packages, and node information; and

placing the subset of the plurality of boot images and the subset of the plurality of software packages in the first storage and the node information in the second storage.

31. (previously presented) A computer-readable storage medium as recited in Claim 21, wherein the master node retrieving the software version information creates the software

version information from the second storage based on functional features included in the request.

32. (currently amended) An apparatus of software loading and initialization in a distributed network of nodes, comprising:

a master node;

a first storage on said master node for persistently storing a plurality of software packages and a plurality of boot images that the nodes in the distributed network will use;

a second storage on said master node for persistently storing software version information and node type information for each node in the network;

means for receiving a request for a boot image and software packages from a node, in the distributed network, that is performing an initial boot;

means for the master node determining, based on the request, software version information of the node to retrieve from the second storage;

means for the master node retrieving the software version information of the node from the second storage;

means for the master node determining, based on the software version information of the node, a boot image of the plurality of boot images and one or more software packages of the plurality of software packages to extract from the first storage;

means for the master node extracting the boot image and the one or more software packages from the first storage; and

means for the master node delivering, to the node, the boot image and the one or more software packages,

wherein said node:

(a) stores the boot image and the one or more software packages in its local persistent storage,

(b) extracts software version information from the one or more software packages and stores the software version information in the local persistent storage,

(c) reboots and executes the boot image stored in the local persistent storage, and

(d) in response to executing the boot image, verifies the software version information with said master node by sending the software version information to the master node;

means for the master node receiving the software version information from the node;

means for the master node determining, in response to receiving the software version information, whether the node has the correct software versions;

means for the master node retrieving, in response to the master node determining that [[if]] said node does not have the correct software versions, correct software packages from the first storage and sending the correct software packages to said node, wherein said node stores the correct software packages in the local persistent storage and completes booting by executing the correct software packages stored in the local persistent storage.

33. (canceled)

34. (previously presented) An apparatus as recited in Claim 32, wherein said node, based on a command from said master node, does not store the one or more software packages in its local persistent storage device, allowing said master node to download test software packages to said node and temporarily run said node using the test software packages,

and wherein when said node reboots, the test software packages will no longer exist on said node.

35. (canceled)

36. (previously presented) An apparatus as recited in Claim 32, wherein if said node has the correct software versions, then said node completes booting by executing the one or more software packages stored in the local persistent storage.

37. (canceled)

38. (previously presented) An apparatus as recited in Claim 32, wherein the master node has the ability to categorize nodes into classes where all of the nodes in a particular class of nodes have the same software configuration and may have differing processor types.

39. (previously presented) An apparatus as recited in Claim 32, wherein each of the one or more software packages contains version information, dependency information, and other metadata information pertaining to software in the package.

40. (previously presented) An apparatus as recited in Claim 32, wherein the boot image is customized for a particular type of node and provides basic low-level communications.

41. (previously presented) An apparatus as recited in Claim 32, further comprising:

means for executing a composite image to create a subset of the plurality of boot images, a subset of the plurality of software packages, and node information; and

means for placing the subset of the plurality of boot images and the subset of the plurality of software packages in the first storage and the node information in the second storage.

42. (previously presented) An apparatus as recited in Claim 32, wherein said means for the master node retrieving the software version information creates the software version information from the second storage based on functional features included in the request.

43. (currently amended) A system for software loading and initialization in a distributed network of nodes, the system comprising:

a master node;

a node in the distributed network;

a first storage on the master node, wherein the first storage persistently stores a plurality of boot images and a plurality of software packages that nodes in the distributed network will use;

a second storage on the master node, wherein the second storage persistently stores software version information and node type information for each node in the distributed network;

one or more processors on the master node;

one or more sequences of instructions which, when executed by the one or more processors on the master node, cause the one or more processors on the master node to perform:

receiving a request for a boot image and software packages from the node that is performing an initial boot;

based on the request, the master node determining software version information of the node to retrieve from the second storage;

the master node retrieving the software version information of the node from the second storage;

the master node determining, based on the software version information of the node, a boot image of the plurality of boot images and one or more software packages of the plurality of software packages to extract from the first storage;

the master node extracting the boot image and the one or more software packages from the first storage;

the master node delivering, to the node, the boot image and the one or more software packages;

~~retrieving correct software packages from the first storage and sending the correct software packages to the node if the node does not have the correct software versions;~~

one or more other processors on the node;

one or more other sequences of instructions which, when executed by the one or more other processors on the node, cause the one or more other processors on the node to perform:

storing the boot image and the one or more software packages in local persistent storage of the node;

extracting software version information from the software packages;

storing the software version information in the local persistent storage;

executing the boot image, that is stored in the local persistent storage, to reboot the node;

in response to executing the boot image, verifying the software version information with the master node by sending the software version information to the master node;

one or more further sequences of instructions which, when executed by the one or more processors on the master node, cause the one or more processors on the master node to perform:

receiving the software version information from the node;

in response to receiving the software version information, determining whether the node has the correct software versions;

retrieving correct software packages from the first storage and sending the correct software packages to the node in response to determining that the node does not have the correct software versions;

one or more additional sequences of instructions which, when executed by the one or more processors on the node, cause the one or more processors on the node to perform:

in response to receiving the correct software packages from the master node, storing the correct software packages in the local persistent storage and executing the correct software packages stored in the local persistent storage to complete booting.

44. (previously presented) A system as recited in Claim 43, wherein the node, based on a command from said master node, does not store the software packages in the local persistent storage device, allowing said master node to download test software packages to said node and temporarily run said node using the test software packages, and wherein when said node reboots, the test software packages will no longer exist on said node.

45. (canceled)

46. (previously presented) A system as recited in Claim [[45]] 43, wherein the one or more other sequences of instructions which, when executed by the one or more other processors, further cause the one or more other processors to perform executing the one or more software packages stored in the local persistent storage to complete booting if the node has the correct software versions.

47. (canceled)

48. (previously presented) A system as recited in Claim 43, wherein the master node has the ability to categorize nodes into classes where all of the nodes in a particular class of nodes have the same software configuration and may have differing processor types.

49. (previously presented) A system as recited in Claim 43, wherein each of the one or more software packages contains version information, dependency information, and other metadata information pertaining to software in the package.

50. (previously presented) A system as recited in Claim 43, wherein the boot image is customized for a particular type of node and provides basic low-level communications.

51. (previously presented) A system as recited in Claim 43, wherein the one or more sequences of instructions which, when executed by the one or more processors, further cause the one or more processors to perform:

executing a composite image, that a user installs on the master node, to create a subset of the plurality of boot images, a subset of the plurality of software packages, and node information; and

placing the subset of the plurality of boot images and the subset of the plurality of software packages in the first storage and the node information in the second storage.

52. (previously presented) A system as recited in Claim 43, wherein the master node retrieving the software version information creates the software version information from the second storage based on functional features included in the request.

53. (previously presented) A method as recited in Claim ] 10, wherein:

the request includes node type information of the node;

the master node determining software version information of the node includes the master node using the node type information of the node to determine the software version information of the node.

54. (previously presented) A computer-readable storage medium as recited in Claim 21, wherein:

the request includes node type information of the node;

the master node determining software version information of the node includes the master node using the node type information of the node to determine the software version information of the node.

55. (previously presented) An apparatus as recited in Claim 32, wherein:

the request includes node type information of the node;

the means for the master node determining software version information of the node includes means for the master node using the node type information of the node to determine the software version information of the node.

56. (previously presented) A system as recited in Claim 43, wherein:

the request includes node type information of the node;

the master node determining software version information of the node includes the master node using the node type information of the node to determine the software version information of the node.